

**REMARKS**

Reconsideration and allowance are respectfully requested in view of the following remarks.

Claims 1-9 and 14-31 are pending in the present application. By this Amendment, claims 1, 4 and 6 have been amended. No new matter has been added.

**Claim Rejections Under 35 U.S.C. § 103**

Claims 1-3, 6, 7, 29 and 31 are rejected under 35 U.S.C. §103(a), on the basis of the Java Remote Method Invocation Specification published February 10, 1997 by Sun Microsystems (identified in the Office Action as "JDOM"), in view of the Jones et al. patent (U.S. 6,557,032, hereinafter "Jones"). The rejection is traversed as follows.

First, there are fundamental differences between the subject matter of claim 1 and the disclosure of JDOM.

According to exemplary embodiments of the present invention, information exchanges among communicating objects in an object-oriented client server system are managed, where the server machine resides in a smart device and the client machine can have access to the smart device via a smart device reader.

JDOM, on the other hand, relates to a Remote Method Invocation implemented in a traditional client-server environment.

Applicants note that since a smart device has limited resources, such as memory, storage, computing power, compared to a traditional server, such as a desk-top computer, an implementation that is practical in a traditional server, may

not be appropriate for a smart device. Therefore, the teachings for a traditional client-server environment, as disclosed in JDOM, should not be directly applied to a client-server environment involving smart devices.

JDOM discloses a Remote Method Invocation using a stub as the client-side proxy and a skeleton as the server-side entity. The stub is an implementation of the remote interfaces of the remote object and forwards invocation requests to the server object via a remote reference layer. The remote reference layer has two cooperating components: the client-side and the server-side components. The client-side component contains information specific to the remote server and communicates via the transport to the server-side component. During each method invocation, the client and the server-side components perform the specific remote reference semantics. For example, if a remote object is part of a replicated object, the client-side component forwards the invocation to each replica rather than just a single remote object. The server-side component handles semantics, such as ensuring atomic multicast delivery by communicating with other servers in a replica group.

JDOM requires implementation of the skeleton and server-side component of the remote reference layer on the server in order to support a Remote Method Invocation. In contrast, claim 1 recites, *inter alia*, sending a process level call request by direct method invocation to the run-time environment of the server machine; and responsive to receipt of said request by the server machine's run-time environment, said run-time environment causing the parameters in the request to become unmarshaled, said remote object to be executed, and the results of the execution to be marshaled.

There are fundamental differences in relying upon skeleton and server-side component of the remote reference layer to support the Remote Method Invocation, as disclosed in JDOM, and relying upon run-time environment to perform functionalities to support the method of information exchanges among objects in server machine and client machine, where the server machine resides in a smart device, and the client machine has access to the smart device, as described in claim 1.

In JDOM, appropriate codes from the skeleton and server-side component of the remote reference layer need to be compiled and linked into the server at runtime. Such compiling and linking require great amount of resources, such as memory or storage. Such compiling and linking, however, may not be desirable on a smart device because of the issue of limited resources.

In contrast, according to exemplary embodiments of the present invention, a run-time environment is used to perform functionalities to support the method of information exchanges among objects in server and client machines. Using run-time environment to perform a functionality on the server side, e.g., the smart device, instead of using different pieces of codes from the skeleton and server-side component of the remote reference layer, demands less resources, and makes an implementation possible, or more optimized, in a client-server environment involving smart devices. Therefore, the differences of using the skeleton and server-side component of the remote reference layer in JDOM, and using run-time environment in the exemplary embodiments of the present invention are fundamental, and should not be dismissed slightly.

In the Response to Arguments section of the current Office Action, it is asserted that any software that runs on an operating system is a run-time environment. Applicants respectfully submit that such assertion is mischaracterizing. In the November 26, 2008 Amendment, it was submitted that a run-time environment is part of a computer's operating system, or software that runs on top of the operating system. In other words, it is closely associated with the basic operations of the computer. To interpret a run-time environment to be any software that runs on an operating system is a run-time environment is over-broadening and incorrect. For example, a piece of code that needs to be compiled and linked at runtime, such as the skeleton disclosed in JDOM, should not be considered as being closely associated with the basic operations of the computer. Even though the skeleton disclosed in JDOM is software, such skeleton is not a run-time environment.

In view of the foregoing, JDOM, disclosing Remote Method Invocation in a traditional client-server environment should not be applied to a client-server environment involving a smart device, as described in claim 1.

In addition, claim 1 recites generating a local object at the client machine based upon interface definition of a remote object resident at the server machine, said local object executable as a proxy to the remote object; said server machine residing in a smart device; and said client machine having access to the smart device via a smart device reader.

Even though JDOM discloses a stub that is an implementation of the remote interfaces of the remote object, JDOM, however, does not disclose that the stub is

generated at a client machine based upon interface definition of a remote object resident at the server machine, where the server machine resides in a smart device; and the client machine having access to the smart device via a smart device reader.

Jones discloses a distributed data processing system, such as a retail store customer transaction network using smart cards, which loads card handling objects in the terminal appropriate to the cards presented. Such distributed data processing system enables smart cards of different types to be handled appropriately.

Jones does not relate to the subject matter of Remote Method Invocation. Therefore, in Jones, the objects loaded into the terminal have nothing to do with proxy objects used in Remote Method Invocation. Instead, the objects loaded into the terminal are merely card handling objects. Therefore, there is no reason to modify the card handling objects in Jones with the client-side stub disclosed in JDOM, which assists Remote Method Invocation.

In view of the foregoing, claim 1 is patentable. Claims 2-3, 6, 7, 29 and 31 are patentable at least because of their dependencies or because they include distinguishing features similar to those of claim 1.

Claims 4, 5, 8, 9 and 30 are rejected under 35 U.S.C. §103(a), on the basis of JDOM and Jones, in further view of prior art described in the present application (identified as "APA").

The alleged APA does not remedy the above-noted deficiencies of JDOM and Jones. Therefore, claims 4, 5, 8, 9 and 30 are patentable.

Claims 14, 16, 18, 19, 21, 23, 24, 26 and 28 are rejected under 35 U.S.C. §103(a), on the basis of Jones and APA, in further view of the DiGiorgio patent (U.S. 6,385,729, hereinafter "DiGiorgio").

In rejecting claim 14, it is acknowledged in the Office Action that Jones and the alleged APA do not disclose generating a local call in response to the single command APDU without the applet having been selected with another command APDU. However, DiGiorgio is cited as an attempt to remedy the deficiencies of Jones and the alleged APA.

DiGiorgio merely discloses a conventional JAVA card that is able to communicate with a computer system by passing APDUs back and forth. DiGiorgio does not teach or suggest generating the local call on the smart device to invoke the method in response to the single command APDU without the applet having been selected with another command APDU, as recited in claim 14.

In view of the foregoing, claim 14 is patentable. Claims 16, 18, 19, 21, 23, 24, 26 and 28 are patentable at least for reasons similar to those for claim 14.

Claims 15, 17, 20, 22, 25 and 27 are rejected under 35 U.S.C. §103(a), on the basis of Jones, APA, DiGiorgio and JDOM.

Since JDOM does not remedy the deficiencies of DiGiorgio, Jones and APA with regard to claims 14 and 24, claims 15, 17, 20, 22, 25 and 27 are patentable at least because of their dependencies.

**C O N C L U S I O N**

From the foregoing, further and favorable action in the form of a Notice of Allowance is respectfully requested and such action is earnestly solicited.

In the event that there are any questions concerning this amendment, or the application in general, the Examiner is respectfully requested to telephone the undersigned so that prosecution of present application may be expedited.

Respectfully submitted,

BUCHANAN INGERSOLL & ROONEY PC

Date: June 17, 2009

By: Weiwei Y. Stiltner  
Weiwei Y. Stiltner  
Registration No. 62979

P.O. Box 1404  
Alexandria, VA 22313-1404  
703 836 6620